

4050 Midterm Project

Shijie An, Xiaoying Lin, Xinchang Liu, Zhen Xu, Yaxuan Yang



Data Wrangling

Merge Dataset

- Combining finance, student progress and statics (30 csv files) in different semesters by **Training and Testing** as df_train and df_test.
- Split out data to finance, progress and statics for further cleaning.
 - Progress_test
 - Finance_test
 - Static_test
 - Progress_train
 - Finance_train
 - Static_train

Cleaning values

No rows with whole row as missing values.



- NA into -1: information are **missing**
 - TransferIntent (Education Objective) & DegreeTypeSought & Final_GPA
- NA into 0: financial records
 - students do **not have scholarship** records, receive \$0 fund; Do not have **income** records
- NA into -2 or 0: **does not apply** or **None**.
 - final_Complete - Students have not completed the program so they do not have degree type
 - Complete CIP - Students have not completed the program so they do not have a graduation field
- NA into **unknown**, for **string** columns
 - Marital.Status, Highest Grade Level, Housing - Student prefer not answered for demographic
- Complete_Dev_Math & Complete_Dev_English
 - Only use two variables 1 means finished, 0 means otherwise (not finished, missing)
- Created variables such as final_Completed (Final Degree) , final_CompleteCIP (Final Major), Final_GPA

Same strategy applied in Testing and Training dataset.

Merge again for df_test_clean and df_train_clean

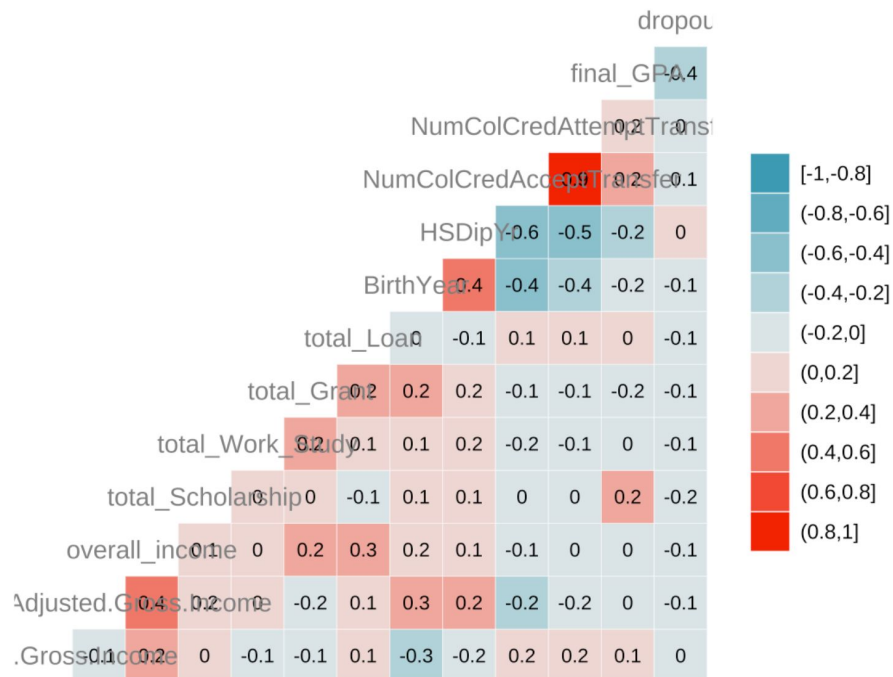
Data Wrangling

- Variables such as major, GPA, degree_Transfer, degree_thought, Complete1, CompleteCIP, complete_DevMath are recorded by semester -> cannot be analyzed among all students because they enrolled in different years -> find the first or last eligible term or both for each person for each variable and created a separate variable instead
- Total amount of each financial aid obtained by students during their college years
- Each income variable has half of the values = 0 -> an overall income.
- Instead of using seven separate race indicators -> race to indicate a student's race directly and drop seven indicators
- Find enrolled_age by subtracting birthyear from cohort_year
- Use floor() on Major variables to indicate general major category
- Check NA and Doesn't Apply proportion
 - Continuous variables with too many missing values -> drop
 - Discrete variables with too many missing values -> test whether those NA has potential meaning
- Little change or variation -> drop

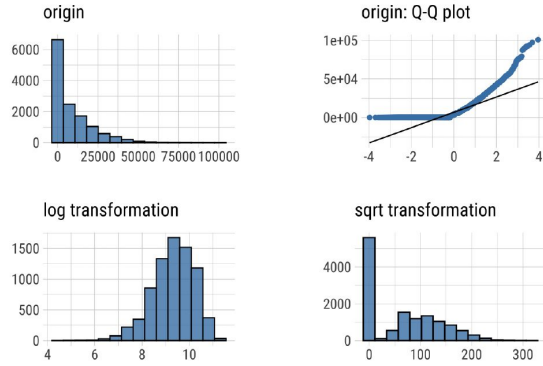


Exploratory Data Analysis & Feature Engineering

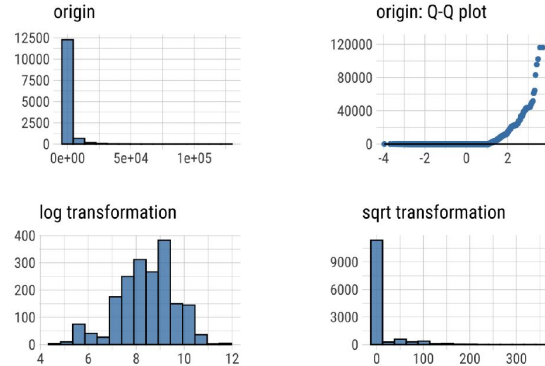
Correlation Plot



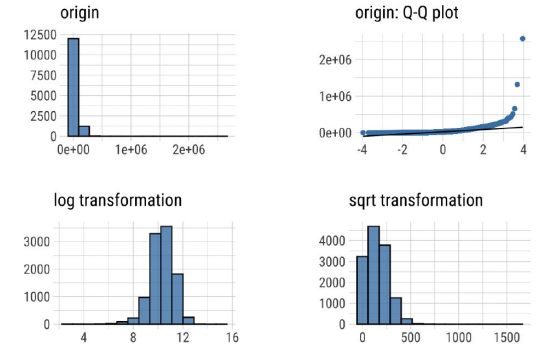
Normality Diagnosis Plot (total_Loan)



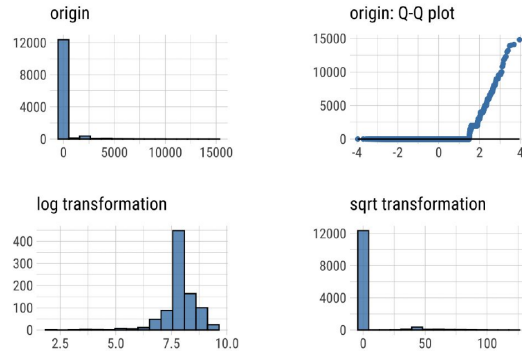
Normality Diagnosis Plot (total_Scholarship)



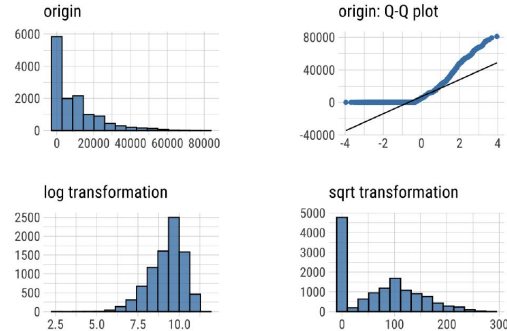
Normality Diagnosis Plot (overall_income)



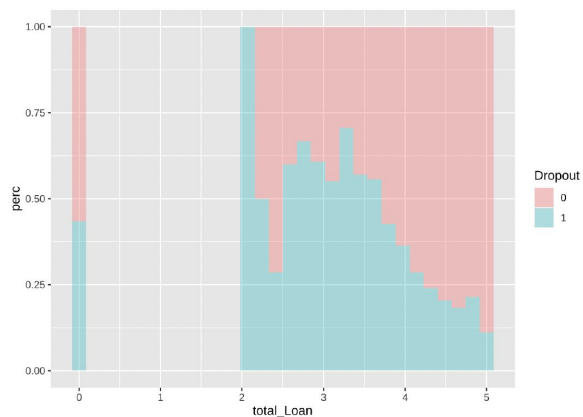
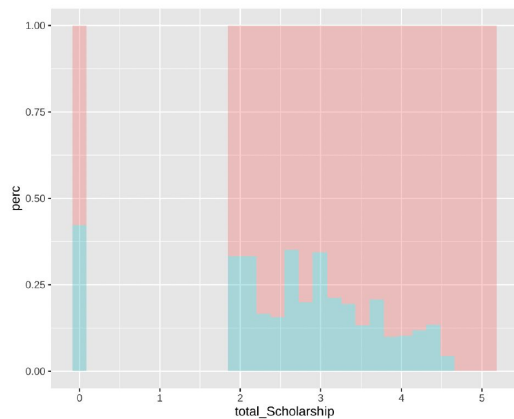
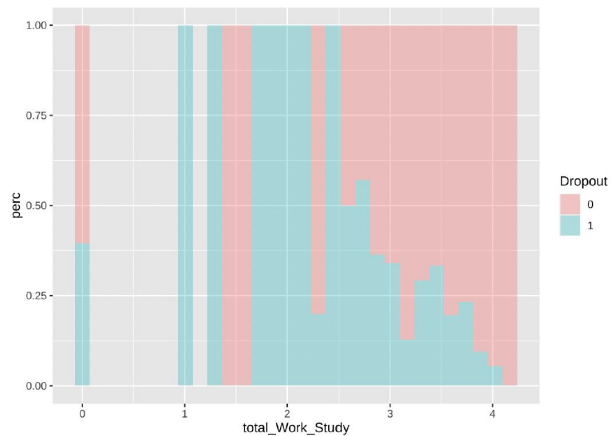
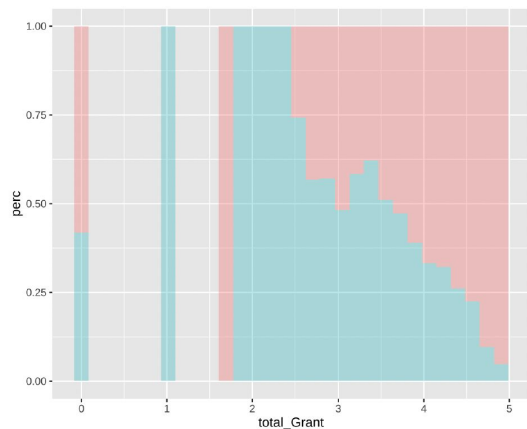
Normality Diagnosis Plot (total_Work_Study)



Normality Diagnosis Plot (total_Grant)



Log Transformation



None of these four variables are normally distributed, so Wilcoxon rank sum test was conducted and results show that all these four variables are significantly correlated with dropout.

Wilcoxon rank sum test (p-value=.001)

Continuous Variables

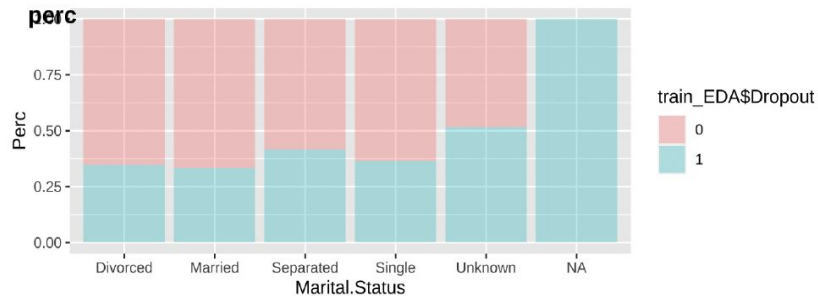
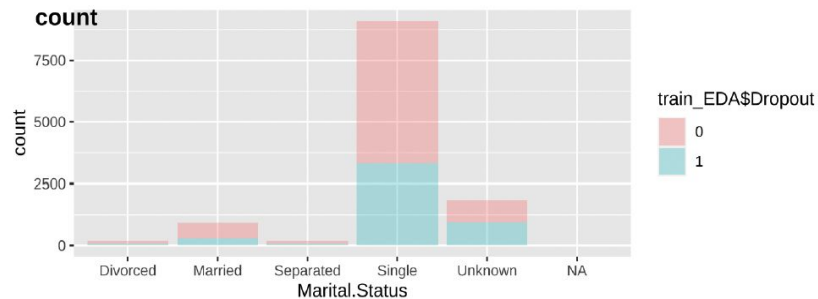
Significant

- Adjusted.Gross.Income, parent.Adjusted.Gross.Income, overall_income, birthyear, NumColCredAcceptTransfer, NumColCredAttemptTransfer, Final_Gpa

Not Significant

- HSDipYr, enrolled_age

Categorical Variables



```
[1] "Marital.Status"
```

Pearson's Chi-squared test

```
data: get(i)
```

```
X-squared = 161.07, df = 4, p-value < 2.2e-16
```

Chi-Square Test

- 4 variables having $p\text{-value} > 0.001$ -> drop
 - Mother.s.Highest.Grade.Level , Gender , BirthMonth , and Complete_Devmath
- The Remaining categorical variables are significant
 - Martial Status, Father.s.Highest.Grade.Level, Housing, EnrollmentStatus, HighDeg, MathPlacement, EngPlacement, GatewayMathStatus, GatewayEnglishStatus, complete_DevEnglish, final_Complete1, first_term_Major1, final_majorOne, race



Models & Results

Support Vector Machine - Linear Kernel

Performance Metrics

Accuracy for testing data: 0.895235

Sensitivity for testing data: 0.9160

Specificity for testing data: 0.8622

Precision for testing data: 0.9135737

Recall rate for testing data: 0.9160021

F-measure score for testing data: 0.9147863

Confusion Matrix and Statistics

Prediction	Reference	
	0	1
0	1723	163
1	158	1020

We choose using SVM because we want to examine whether classes are nearly separable or not. Based on the correlation plot we made in EDA, we assumed that it is non-linear. For non-linear boundaries, kernel SVMs computations are less expensive than Logistic Regression.

Support Vector Machine - Polynomial Kernel

Performance Metrics

Accuracy for testing data: 0.9001305

Sensitivity for testing data: 0.9266348

Specificity for testing data: 0.8579882

Precision for testing data: 0.9120879

Recall rate for testing data: 0.9266348

F-measure score for testing data: 0.9193038

Confusion Matrix and Statistics

Prediction	Reference	
	0	1
0	1743	168
1	138	1015

Support Vector Machine - Radial Kernel

Performance Metrics

Accuracy for testing data: 0.8962141

Sensitivity for testing data: 0.9149389

Specificity for testing data: 0.8664413


Precision for testing data: 0.9159127

Recall rate for testing data: 0.9149389


F-measure score for testing data: 0.9154255

Confusion Matrix and Statistics

Prediction	Reference	
	0	1
0	1721	158
1	160	1025



Based on the results of three SVM Kernel Models, for both accuracy and F score, Polynomial Kernel has the best performance. As a result, the decision boundary is non-linear. However, since we need to pay more attention to students who will drop out and exercise necessary interventions, we need to focus on specificity (negative=1 in models). In that perspective, Radial Kernel performs the best.



Naive Bayes

Performance Metrics

Accuracy for testing data: 0.7428

Sensitivity for testing data: 0.9048

Specificity for testing data: 0.4852

Precision for testing data: 0.7365

Recall rate for testing data: 0.9261

F-measure score for testing data: 0.8120

Confusion Matrix and Statistics

Prediction	Reference	
	0	1
0	1879	1174
1	2	9

Naive Bayes is good at solving classification problems, and it can be used for high-dimensional training dataset. It handles both discrete and continuous data, and highly scalable with the number of predictors and data points.

Logistic Regression

Performance Metrics

Accuracy for testing data: 0.8871

Sensitivity for testing data: 0.9261

Specificity for testing data: 0.8250

Precision for testing data: 0.8954

Recall rate for testing data: 0.9261

F-measure score for testing data: 0.9097

Confusion Matrix and Statistics

Prediction	Reference	
	No	Yes
No	1737	203
Yes	144	980

Logistic regression has some irreplaceable advantages such as the fitting effect, and it usually has a calculation efficiency that is better than SVM or random forest.

Decision Trees

Performance Metrics

Accuracy for testing data: 0.8685

Sensitivity for testing data: 0.9256

Specificity for testing data: 0.7777

Precision for testing data: 0.8688

Recall rate for testing data: 0.9256

F-measure score for testing data: 0.8963

Confusion Matrix and Statistics

Prediction	Reference	
	0	1
0	1741	263
1	140	920

Decision trees used tree-based partitions to handle both continuous and categorical variables, so it can provide good classification for the data set we are working on.

Random Forest (Ensemble)

Performance Metrics

Accuracy for testing data: 0.9238

Sensitivity for testing data: 0.9394

Specificity for testing data: 0.8990

Precision for testing data: 0.9367

Recall rate for testing data: 0.9394

F-measure score for testing data: 0.9380

Confusion Matrix and Statistics

Prediction	Reference	
	0	1
0	3535	239
1	228	2127

Similar mechanism with decision tree to provide a good classification for continuous and categorical variables. But the random forest algorithm performs better in accuracy when predicting outcomes than the decision tree algorithm.

XGBoost (Ensemble)

Performance Metrics

Accuracy for testing data: 0.9223

Sensitivity for testing data: 0.9442

Specificity for testing data: 0.8876

Precision for testing data: 0.9303

Recall rate for testing data: 0.9442

F-measure score for testing data: 0.9372

Confusion Matrix and Statistics

Prediction	Reference	
	0	1
0	1776	133
1	105	1050

XGBoost is an ensemble algorithms under the [Gradient Boosting](#) framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems(classification and regression) in a fast and accurate way.



Problems

When first cleaning the data, we used the train dataset extracted from the whole dataset. Then, we found out that we needed to clean the test dataset again and merge the train and test dataset together. As a result, when we later wrangled and analyzed the data, we used train dataset to analyze and performed feature engineering on the dataset to avoid repetitive actions.

Absolute File Path

```
trainEDA<-read.csv("/Users/xinchangliu/Dropbox/Mac/Desktop/train_AFTER_EDA.csv")
```

Relative Path

```
df_test<-read.csv("Student Retention Challenge Data/Test Data/TestIDs.csv")
```

